Design and Development of Certification Compliance Tool for Airborne Systems

Anusha B K
Master of Engineering
Embedded Systems
School of Information Sciences
Manipal University
Manipal, India
anu06bk@gmail.com

Dr. Manju Nanda
Aerospace Electronics &
Systems Division, CSIR
National Aerospace
Laboratories
Bangalore, India
manjun@nal.res.in

J Jayanthi
Aerospace Electronics &
Systems Division, CSIR
National Aerospace
Laboratories
Bangalore, India
jayanthi@nal.res.in

***Abstract*— Certification compliance check for airborne software is very critical as it aids in the certification of the software. Since this compliance check is performed manually which is time-consuming and erroneous, an in-house developed Certification Compliance Tool (CCT) helps in checking the compliance as per RTCA DO-178B/C and generate artifacts depicting the magnitude of compliance. In order to generate the magnitude of compliance for the artifacts with respect to the Civil Aerospace Certification standard, RTCA DO-178B/C, an effective parsing technique is required to be incorporated to parse the artifact/s and generate compliance metric for the artifact/s.**

In this paper we propose a novel approach used in the design and development of an effective and efficient parsing technique incorporated in the indigenous software tool CCT used for compliance check. The tool checks the ratio of compliance of the artifacts generated across various phases of Software Development Life Cycle (SDLC) process involved in the development of Safety-Critical software as per RTCA DO-178B/C. The indigenous tool accepts these artifacts as inputs and based on the software criticality level, it analyzes the compliance of these artifacts with the guidelines provided and recommended by RTCA DO-178B/C. The output of the tool provides the percentage of compliance of the artifacts that helps in accessing the Certification capabilities of the developed software.

The percentage of compliance predicts the acceptance or rejection probabilities of the software being certified by the Certification Agency. The certification parser is developed using Python modules like Pywin32, Pypdf parsers and different approaches for Natural language processing using Python Natural Language Toolkit (NLTK). The in-house tool replaces the manual effort by an individual/s which may be erroneous and impact the time-schedule, which compromises the software safety. The integration of the tool with commercial tools will help in analyzing the report/ documentation content with respect to the certification.

Index Terms*— *Safety Critical System, Certification Compliance Tool (CCT), Software Development Life Cycle (SDLC), Natural Language Toolkit (NLTK), Pywin32, Pypdf, Compliance Percentage.

I. INTRODUCTION

Certification is one of the most important aspects considered by various industries that recommend the use of legit software and its components in design and development of complex embedded systems of various Safety-Critical Control Systems [1]. The certification authority legally recognizes the software product that complies with the standards specified as per RTCA DO-178B/C [9]. Certification can be applied to individuals like operator or user of a system, organizations, standards, processes or final products, experts, tools or methods. Software certification plays a vital role in demonstrating the reliability and safety of software systems [2]. Safety is a critical criterion in the development of Safety-Critical systems. Safety-Critical systems are those systems whose failure can result in injury/loss of life from system failures, collateral damage to property, damage to resources and reputations, or damage to the environment. The examples of Safety-Critical systems are aircraft flight control, nuclear reactor control, medical treatment systems, missile system control software etc.,

In order to avoid the Safety-Critical system failures, the U.S. government along with globally placed international agencies have proposed a number of standards, guidelines, and reports related to certification and other aspects of software assurance, such as qualification, or validation, licensing, in their specific areas of interest [1]. DO-178B/C is one such guidance document for civil aviation, developed by Radio Technical Commission for Aeronautics (RTCA- Inc)., provides guidelines with justifications in assuring software developed for airborne systems .

The purpose of Certification of Compliance is to improve the safety of the critical systems, to increase the awareness of the implications of system performance on safety, to enforce minimum standards of design and manufacture within the relevant industry, to encourage a structure of professional responsibility [8].

In this paper, we propose an approach to automate the process of checking the artifact compliance with respect to the RTCA DO-178B/C standard. The artifacts are the results of different phases (Requirements, Architecture, Design, Integration and Testing) of **S**oftware **D**evelopment **L**ife **C**ycle (SDLC) process. The artifacts of different formats are efficiently parsed using Pypdf and Pywin32 parsers. The extracted data is pre-processed using Natural Language Processing with Python

Natural Language Toolkit (NLTK). The pre-processed data is analyzed with the standards specified in RTCA DO-178B/C, with the documents being validated semantically similar words using Wordnet. Thus, the Compliance Percentage for artifact is generated.

This paper is organized as follows: Section II deals with Literature survey and introduction to the existing system. Section III introduces to the approach being incorporated in the development of the tool. Section IV gives an overview of the processing and comparison techniques involved in CCT. Section V analyzes the results obtained from the proposed approach. Section VI concludes the work carried out in this paper with outline of the future scope.

II. Literature Survey

Safety Standards are required by the organizations, authorities, or law during the development and certification of Safety Critical systems [8]. Standards provide a set of guidelines as objectives that are to be satisfied during the Software Development Life Cycle. For safety functions of Safety Critical systems, all standards have a risk oriented approach specifying Risk Reduction Factors and Probability of Failure on Demand [24]. Standards give confidence in achieving compliance. The classification of Safety standards for various industries [7]:

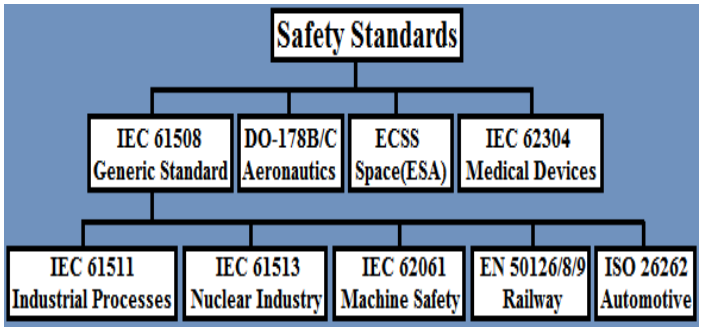

Fig 1: Safety Standards for various industries

Radio Technical Commission for Aeronautics (RTCA, Inc.) which is a private association of aeronautical organizations includes NASA, DOD, FAA and other government agencies, airline operators, aircraft equipment suppliers, airline manufacturer, and various pilot associations [3], seeks sound technical solutions to problems involving the application of telecommunications and electronics to aeronautical operations [10]. The objectives of RTCA includes ensuring the reliability and safety of airborne systems, developing the minimum operational performance requirements for the document-specific systems, developing the guidelines for the use of regulatory authority, enabling the teamwork among the worldwide aviation community by providing administrative and logistics resources [11].

DO-178 described in RTCA DO-178B/C document titled "Software Considerations in Airborne Systems and Equipment Certification" is developed to establish software considerations for users, developers and installers, the microcomputer techniques that are used to implement aircraft complex hardware and software components [10]. The purpose of DO-178B/C is to provide guidelines for the production of software for airborne systems and equipment that performs its intended function with a level of confidence in safety that complies with airworthiness requirements [4].

DO-178C also defines five levels of design assurance:

1. **Level A → Catastrophic failure** condition for the aircraft
   - Software whose anomalous behavior would prevent continued safe flight and landing or loss of aircraft and occupants.
   - Flight surface controls, engine controls, etc.
2. **Level B → Hazardous** failure condition for the aircraft
   - Software whose anomalous behavior would cause large reduction in safety margins, serious/fatal injuries to occupants or higher crew workload.
   - Primary Flight Displays, Cabin Pressurization, etc
3. **Level C → Major** failure condition for the aircraft
   - Software whose anomalous behavior would result in significant reduction in safety, discomfort to occupants, or significant increase in crew workload
   - Flight Management Systems, COMM, NAV, DATALINK, etc.
4. **Level D → Minor** failure condition for the aircraft
   - Software whose anomalous behavior does not significantly reduce aircraft safety and involves crew actions well within capability.
   - Transponders, cabin lighting, etc.
5. **Level E → No effect** on the safe operation of the aircraft
   - Software whose anomalous behavior does not affect operational capability and doesn't result in an increase in crew workload
   - In-flight entertainment, satellite phone, etc.[19][25].

A. Existing System

The existing approach involves manual checks for compliance of the artifacts with the RTCA DO-178B/C standards. The verified and validated artifacts, checked for compliance, are sent to certification authority for the certification of the software/hardware component. The rejection possibilities will be higher if these artifacts are not compliant with the standards specified as per RTCA DO-178B/C. The manual process is a time consuming process and requires lot of effort in identifying any faults in these artifacts. For artifacts which are very dense this process is difficult and laborious.

III. Approach Being Incorporated in the Development of the Tool

With the existing approach there is a need to automate the document compliance with standards as per RTCA DO-178B/C. Model-Based Software Engineering approach stress on the need for compliance checks and an efficient tool is very much critical for the artifacts generated by these tools. These artifacts that are compliant are complied with appropriate feedback provided by the **C**ertification **C**ompliance **T**ool (CCT). A compliance check automation tool is developed to check the compliance of the artifacts with the standards specified as per RTCA DO-178B/C.

Based on the software criticality levels, the Safety-Critical software systems are classified into Level A (most critical), B, C, D and E (General). Each Criticality level has Design Workflow, Verification and Validation Workflow objectives. The approach given in this section explains the process of Software Development Life Cycle (SDLC) with respect to the criticality levels and their techniques. The various Design techniques are Conventional, Object Oriented, Model Based and Model Based Object Oriented. The Verification and Validation techniques are Conventional, Model Based, Object Oriented, Formal Method, Model Based Object Oriented, Model Based Formal Method and Formal Method Object Oriented [3][21].

The Software Development Life Cycle phases are Requirement, Software Architecture (High-level Requirements), Software Design (Low-level Requirements), Integration & Testing (Verification & Validation). Each phase use different techniques, methodologies and tools with the definition of a set of activities that needs to be completed successfully with the artifacts generation from the subsequent phases. Thus, artifacts that are generated to check for its compliance percentage with standards as per RTCA DO-178B/C guidelines for each phase in SDLC.

The artifacts generated by the phase specific tools are saved in either *.pdf* or *.docx* format. These artifacts are then input to CCT, where they are parsed using Python Pypdf [23], Pywin32 [22] respectively. Further on with the help of Natural Language Processing effective data extraction is performed and analyzed further for the compliance with respect to the standards as per RTCA DO-178B/C. With the proposed approach, the compliance check is automated and a percentage of compliance is provided for the suitable artifacts from the respective phases of SDLC.

A. Proposed Workflow

1. The UI of CCT detailed using HTML and CSS is used to define the criticality levels (level A, B, C, D & E) of the Software Development Life Cycle (SDLC) process.
2. Select the Workflow (Design / Verification & Validation) to understand the objectives of different phases of SDLC process as per RTCA DO-178B/C.
3. Select the appropriate phase of SDLC and input the artifacts in *.pdf/.docx* format obtained based on various methodologies, techniques and tools used for each phase of SDLC.
4. Parse the artifacts using Pypdf, Pywin32 for *.pdf* and *.docx* artifacts parsing respectively.
5. Apply Natural Language Processing on the parsed artifacts using Python Natural Language Toolkit (NLTK).
6. Analyze by comparing the Table of Contents (TOC) and keywords of the parsed artifacts with RTCA DO-178B/C standard guidelines for TOC and keywords.
7. Generate the Compliance percentage (output).

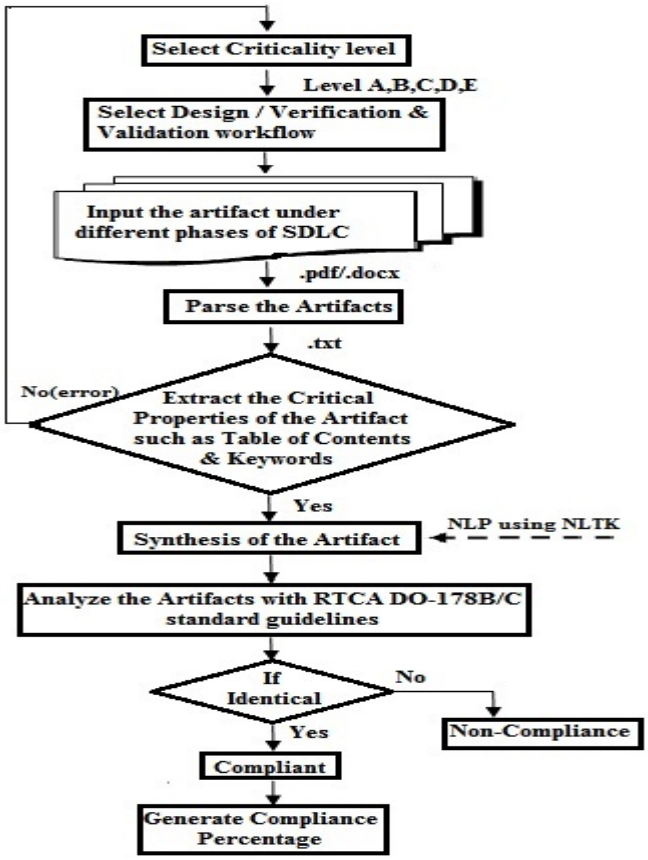

Fig 2: Workflow of the proposed approach

IV. Processing and Comparison Techniques Involved in CCT

The artifacts are browsed from specific directories and provided as inputs to CCT. The browsed directory path which has the filename along with it is parsed by the parser to analyze the name of the file with particular standard documents, this helps in guiding the parsing process and sequences the extraction of TOC & keywords based on the filename type.
Ex: Sdd.pdf (Software Design Description), Srd.docx (Software Requirements Data).

A. Parsing modules

The artifacts in the form of *pdf* and word file formats are considered as binary files, making them more complex for synthesis than the plaintext files. Parsing a *.pdf/.docx* file consumes more time and memory. To parse all the *pdf &* word files and extract text from each page, there are Python modules that make it easy to interact with *.pdf* and *.docx* documents.

The parsers provide simple, unambiguous and powerful mechanisms to extract complex structured text content from pdf and word file formats. A parser implementation includes structural information synthesis from the extracted content which is helpful to judge the relevance of different parts of the parsed artifact.

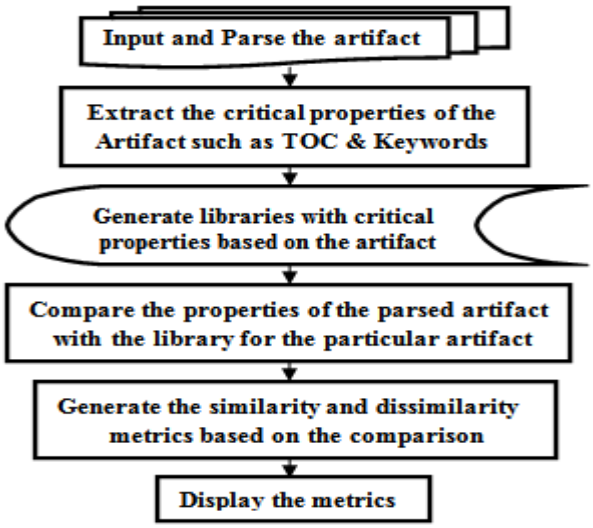

Fig 3: Workflow of processing and comparison techniques

1) Pypdf and Pywin32

Pypdf and Pywin32 are the Python libraries used to extract the data from *pdf* and word files respectively. The parsers focus entirely on analysis, getting data and capable of:

- Extracting the structured document information like title, author and table of contents etc.,
- Split a range of pages from *pdf/docx* document into separate single-paged *pdf/docx* documents
- Extract exact location of text in a page.
- Merging and cropping the pages of *pdf* and word file.
- Rotate a range of pages of *pdf/docx* document by a specified angle.
- Encrypting and decrypting the *pdf/docx* files.
- Converting *pdf* and word files to *.txt* file

The *pdf* and word documents are parsed by Pypdf and Pywin32 respectively to iterate and syntactically analyze all pages of the document for finding and extracting the structured text content.

The Python code for doing the above:

For PDF

```
def getPDFContent(path):
    content = ""
    pdf = pyPdf.PdfFileReader(file(path, 'rb'))
    for i in range(0, pdf.getNumPages()):
    content += pdf.getPage(i).extractText() + '\n'
```

For DOCX

```
app=win32com.client.Dispatch('Word.Application')
app.Visible = False
in_file=os.path.abspath(rootdir+"\\"+filename+""+fileextn)
doc = app.Documents.Open(in_file)
content = doc.Content.Text
text = doc.Range().Text
```

The critical properties of the artifact such as TOC and keywords are extracted and pre-processed. The Pre-processing of the extracted critical properties of the artifact is done by Natural Language Processing using Python Natural Language Toolkit (NLTK).

B. NLP – Natural Language Processing

Natural Language Processing (NLP) is a technique to analyze, manipulate and interact between the computer and natural language or speech in area of application and research. A variety of tools have been developed to perform Natural Language Processing. One such tool is Python Natural Language Toolkit (NLTK) [13]. The Applications of Natural Language Processing are Text processing, Information extraction, Semantic Analysis, Language Modeling, Sentiment Analysis, Transliteration, Document Similarity, Automatic Summarizing, Machine translation, Opinion Mining, Question Answering, Discourse Analysis, Document classification.

C. NLTK

Natural Language Toolkit is a platform for building programs using Python Programming language to interact with the natural language data. NLTK was designed with four primary goals: Simplicity, Consistency, Extensibility and Modularity. Natural Language Toolkit provides interfaces to 50 corpora and wordnet (lexical resource), along with text processing libraries for tokenization, tagging, chunking, classification, parsing, stemming and semantic reasoning [14].

The parsed artifact undergoes pre-processing step as follows (Fig 4):

1. Collapsing the unwanted whitespaces between the text content.
2. The punctuations and digits from the extracted content are eliminated.

3. Content of text is converted to lower case.
4. Text content segmentation is done using sentence tokenization.
5. Removing the list of stopwords like a, an, the, if, in, etc..,
6. Assigning each word in a sentence the part of speech that it assumes in that sentence by Part-of-Speech tagging.
7. Extracting the Noun phrases using Regular expressions.

Thus, Libraries with critical properties are generated for the artifacts and the standard documents. The critical properties of the parsed artifacts are compared with the libraries of the standard artifact which is as per RTCA DO178B/C.

D. Comparison Module

- The artifacts are analyzed with the RTCA DO-178B/C standard documents.
- The artifacts are categorized under its criticality level and technique used.

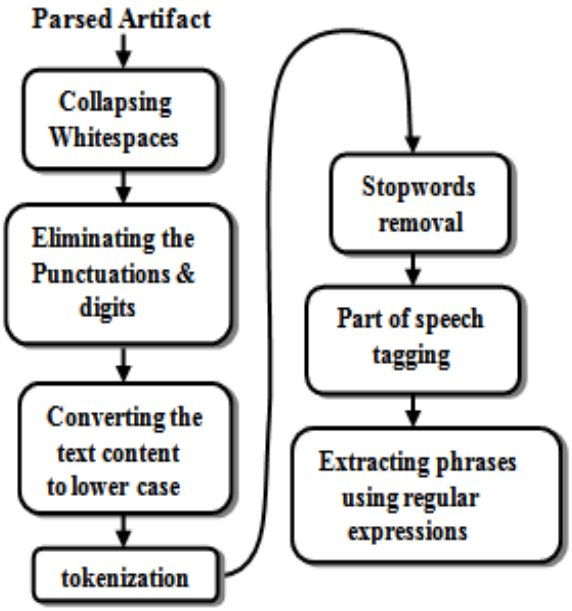

Fig 4: Steps in Pre-processing

- To avoid the criticality issue the artifact of Level A should not be compared with any other Software Levels (Level B, C, D & E).
- The similarity of critical properties of the artifacts with the standard documents is computed using Wordnet Semantic similarity.
- The extracted phrases of an artifact which are not similar with the standard document phrases are directed to the list of the non-compliance under the particular artifact.
- Semantic similarity using Wordnet will give the probability value of similarity of two words.

1) Wordnet and Semantic similarity

Wordnet is a lexical resource, consists of large database of English. The Parts of Speech such as noun, verbs, adjectives and adverbs are grouped into sets of synonyms, also called as synsets. Each one of these explains a distinct concept [17][18]. Synsets are linked by a complex network of lexical relations. Given a particular synset, we can traverse the WordNet network to find synsets with related meanings. Wordnet is used to find the semantic similarity between sentences, words and paragraphs using the hierarchical structure among the hyponym/hypernym and meronym/holonym relations [15].

Semantic similarity using Wordnet are widely used in Natural Language Processing and the retrieval of information. The semantic similarity between the words, sentences and paragraphs has an important role in many scientific research areas such as Cognitive Science, Linguistics, Artificial Intelligence and Knowledge Engineering [16]. Semantic similarity measures are grouped into four classes: information content based measures, hybrid measures, path length based measures and feature based measures [6].

For a particular document,

- Average probability of an artifact = Average of all compared probability values.
- Compliance percentage of an artifact = (Average probability)*100

The following Python code shows the similarity and the dissimilarity of an artifact when compared with the standard document.

```
comp = [ ]
noncomp = [ ]
result = [ ]
maxvalue = 0
h = 0
for i in lib1:
   maxvalue=0
   for j in lib2:
      if(i.lower()==j.lower()):
            maxvalue=100
            print maxvalue
            else:
              try:
                 synsi=wordnet.synsets(i)
                 synsj=wordnet.synsets(j)
                 i1=synsi[0].name()
                 j1=synsj[0].name()
                w1=wordnet.synset(i1)
                w2=wordnet.synset(j1)
                s=w1.wup_similarity(w2)
                h=s*100
```

```
if(h>maxvalue):
        maxvalue=h
        maxname=j
    result.append(maxvalue)
for i in range(0,len(result)):
    if(result[i]>90):
        comp.append(lib1[i])
    else:
        noncomp.append(lib1[i])
```

V. RESULTS

The implementation is done using Python parsing modules like Pywin32, Pypdf. The pre-processing step and comparison of the parsed artifact is carried with the help of Natural Language Processing using Python Natural Language Toolkit. The proposed approach helps to find the reason for the Non-Compliance of generated artifacts with the standard documents in early stage. The requirements like safety, efficiency, time and accuracy are managed and satisfied.

Fig 5: Front end of the Certification Compliance tool

Based on the Software criticality, the classification of safety critical software systems are shown in Fig 5. On Clicking the required Level, the tool flows with display of Design Workflow, Verification and Validation Workflow as shown in Fig 6.

The artifacts are given as input with respect to different phases of Software development Life Cycle under particular criticality level and the technique used as shown in Fig 7.

Fig 8 shows the compliance percentage of an artifact sdd.docx which is 94.468932% accurate when compared with the standard document and also shows the reasons for its non-compliance. Table 1 shows the generated compliance percentage for various artifacts.

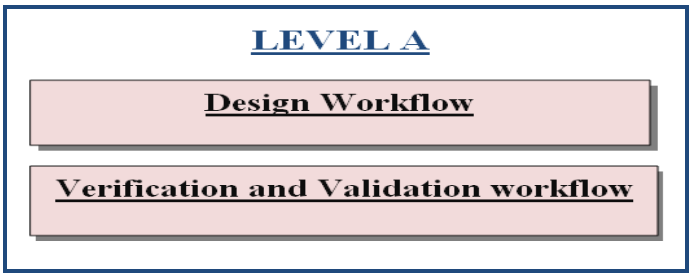

Fig 6: Display of design, verification and validation workflow

.

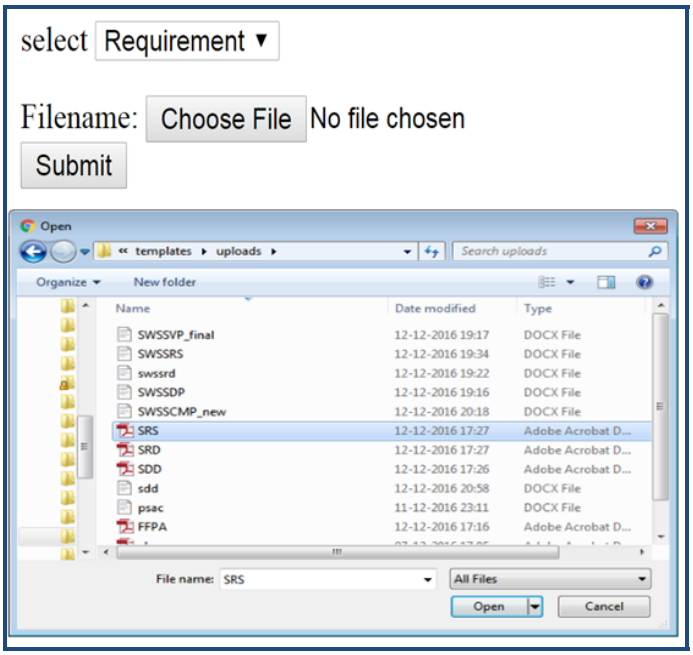

Fig 7: Certification Compliance tool takes generated artifacts as input

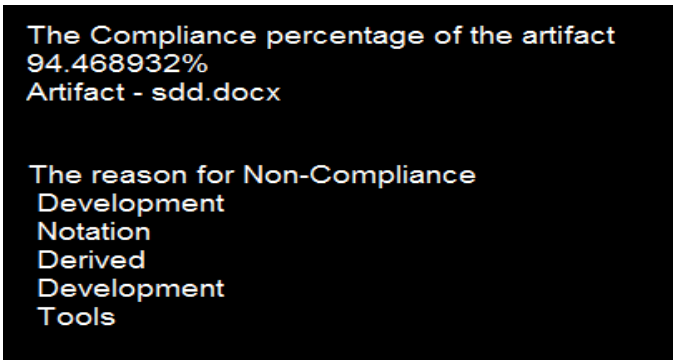

Fig 8: Generated compliance percentage of an artifact and reason for non-compliance

TABLE 1

Filename	Compliance %
psac.pdf	98.823529
sdd.docx	94.468932
svcp.docx	100.00000
swssdp.docx	90.746753
scmp.pdf	96.311367
srs.docx	96.311367

VI. Conclusion and Scope for Future Work

The process of compliance check automated with the help of Certification Compliance Tool (CCT) can be performed efficiently and effectively. The work proposed in this paper aids in providing confidence on compliance checks with respect to standards as per RTCA DO178B/C. It generates compliance percentage for the artifacts upon comparing with RTCA DO-178B/C standard guidelines. It makes an attempt to show the role of parsers hiding the complexity of different file formats and allows even huge artifacts from different phases of Software Development Life Cycle to be parsed and pre-processed without any difficulty and involvement of any laborious manual process. The future work includes adding more intelligence to the tool by extracting more critical properties from the whole document text content and analyzing with standard documents to obtain accurate Compliance Percentage.

Acknowledgment

The authors would like to thank the Director of CSIR-NAL, Bengaluru, for supporting this work.

References

[1] Kornecki, Andrew, and Janusz Zalewski. "Software certification for safety-critical systems: A status report." Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on. IEEE, 2008.

[2] Nelson, Stacy. "Certification processes for safety-critical and mission-critical aerospace software." (2003).

[3] Jacklin, Stephen. "Certification of safety-critical software under DO-178C and DO-278A." Infotech@ Aerospace 2012. 2012. 2473.

[4] Holloway, C. Michael. "Making the implicit explicit: Towards an assurance case for do-178c." (2013)..

[5] Dodd, Ian, and Ibrahim Habli. "Safety certification of airborne software: An empirical study." Reliability Engineering & System Safety 98.1 (2012): 7-23.

[6] Farkiya, Alabhya, et al. "Natural Language Processing using NLTK and WordNet."

[7] N, Manju, et al. "A Framework for the Software Aspects of the Safety Certification for Indigenously Developed Aircraft Systems." International Journal of Advanced Research in Electrical, Electronics and Instrumentation Energy, Research and Reviews, Dec.2013.

[8] Traussnig, Robert, and Holger Giese. "Safety-Critical Systems: Processes, Standards and Certification." Seminar "Analysis, Design and Implementation of Reliable Software. 2004.

[9] "Certification." Micrium,www.micrium.com/certification/why-safety-critical-certification.

[10] Spitzer, Cary R. The Avionics Handbook. Boca Raton, CRC Press, 2001.

[11] "Radio Technical Commission for Aeronautics." Wikipedia, Wikimedia Foundation,17Apr.2017,en.wikipedia.org/wiki/Radio_Technical_Commission_for_Aeronautics. .

[12] Knuuttila,Tarja. "Chapter Twenty One Language Technological Models As Epistemic Artefacts: The Case Of Constraint Grammar Parser". Academia.Edu,2017,http://www.academia.edu/1583180/Chapter Twenty One Language Technological Models As Epistemic Artefacts The Case Of Constraint Grammar Parser.

[13] Jaganadh Gopinadhan, Sr. Applied Data Scientist at Cognizant Follow. "Natural Language Processing." LinkedIn SlideShare, 9 Oct. 2012, www.slideshare.net/jaganadhg/natural-language-processing-14660849.

[14] "Natural Language Toolkit¶." Natural Language Toolkit — NLTK 3.2.4 Documentation, www.nltk.org/.

[15] Segeblad, Jesper. "WordNet – Structure and Use in Natural Language Processing."WordNet – Structure and Use in Natural Language Processing,pdfs.semanticscholar.org/2068/34df3166b5a386351e8c4d069e64e3b7105e.pdf.

[16] Gole,Sagar. "Understanding words similarity/Relatedness using Wordnet",20 April. 2015, http://blog.thedigitalgroup.com/blog/understanding-words-similarityrelatedness-using-wordnet/

[17] University, Princeton. "What Is WordNet?" Princeton University, Trustees of Princeton University © 2017, 17 Mar. 2015, wordnet.princeton.edu/wordnet/.

[18] Simpson, Troy, and Thanh Dao. "WordNet-based semantic similarity measurement." The Code Project. com (2005).

[19] "An overview of the new DO-178C guidelines and what it means for your next test application".2012, ftp://ftp.ni.com/pub/branches/uk/ats_2013/What_DO_178C_Means_for_Your_Application.pdf

[20] "AdaCore- GNAT PRO Safety-Critical".2014, http://www.adacore.com/gnatpro-safety critical/avionics/ do178c/

[21] Spitzer, Cary R., and Cary Spitzer, eds. Digital Avionics Handbook. CRC press, 2000.

[22] "Pywin32 - Python Wiki". Wiki.Python.Org, 2017, https://wiki.python.org/moin/PyWin32.

[23] "Pypdf 1.13 : Python Package Index". Pypi.Python.Org, 2017, https://pypi.python.org/pypi/pyPdf/1.13.

[24] What Is the Position of Standards Regarding Formal Methods in My Industry Segment? - Evidence on Formal Methods Uses and Impact on Industry, www.fm4industry.org/index.php/What is the position_of_standards_regarding_formal_methods_in_my_industry_segment%3F.

[25] Circular, Advisory. "Federal Aviation Administration." US Department of Transportation, AC 150/5320 (1988).

[26] Software Considerations in Airborne Systems and Equipment Certification. Washington, D.C., RTCA, Inc., 1992.